ISSN 0956-9979 JANUARY 1995

VIRUS BULLETIN

THE INTERNATIONAL PUBLICATION ON COMPUTER VIRUS PREVENTION, RECOGNITION AND REMOVAL

IN THIS ISSUE:

• **The 1995 Scanner Comparative.** Twice a year, *Virus Bulletin* puts a collection of virus scanners through an increasingly difficult set of tests, which are aimed at discovering which vendors can keep up with the pace. With the number of viruses now over 5000, who (if anyone) is lagging behind?

• **The many faces of protection.** A scanner may be the most popular form of anti-virus software, but is it the best? A short review of other anti-virus techniques, and their strengths and weaknesses, is given on p.12.

• **Viruses and networks.** A new way of securing a network makes its *VB* debut this month: the *Vi-SPY Universal Network Installable Module* (NIM) from *RG Software*. Discover how it differs from an NLM, and where its use is appropriate.

CONTENTS

EDITORIAL

Hey, Good-lookin'!

It is only natural to be attracted by good looks, no matter what the object in question. Cars, computers, even tin-openers… all are designed to be as æsthetically pleasing as possible, in order to woo a prospective purchaser. However, when it comes to anti-virus software, how much should the user interface affect the purchasing decision? In a nutshell, does a pretty front-end make up for lack of detection capability?

Anti-virus software comes in all shapes and sizes (although an increasing amount seems to be owned by *Symantec* - perhaps this will become a new industry standard?), and user interfaces range from nil to stunning. However, evaluating the 'usability' of a scanner is highly subjective, and prone to preconceptions about what makes a 'good' product. *Virus Bulletin* has always taken a rather stern view of 'pretty' products, rightly judging user interface as less critical than virus detection. However, with the Form virus still topping the virus prevalence charts, it would seem that there is a good argument that users are either not using anti-virus software at all, or are using it incorrectly. Therefore, a careful look at the true usability of anti-virus software is long overdue.

"it soon became abundantly clear why the scanner is the universally-accepted front line of defence"

Always thirsting for a new story (and the chance of a bottle of wine on expenses...), a handful of friends were bribed with the offer of a meal and a drink, given a pile of installation disks and manuals, and let loose on the Editor's home machine. The resulting chaos was almost amusing enough to make up for the hours spent trying to restore the machine to its original condition.

'Nice,' commented one, 'very easy to use'. 'Nice blue colour' was another remark. But what of functionality? Only the most technically-able was interested in whether any viruses were detected or not: the remainder of the group were more worried about speed. The only technical question was one particularly well-timed comment pointing out that there were only 60 seconds in a minute, so why should one look for 62 - was this part of new EC legislation on time management? Detection ratios? TSR size? DES-Encrypted CRC checksumming? Nobody cared.

Certain products did extremely poorly in the comprehensibility stakes. Comments like 'very nice, but is it infected or not?' began to be thrown around, and it soon became abundantly clear why the scanner is the universally-accepted front line of defence: it displays a simple 'everything is okay' message, which is easy to understand.

After a fun-filled hour, everyone tired of the experiment (not surprising, given the rather measly payment offered in return), but one fact had been brought out clearly: there is a large gap between the command line interface, which is the techie's personal favourite, and the pretty pictures which go down well with the user. The technical aspects were quickly forgotten. Would anyone clean boot? Probably not - the 'it won't happen to me' syndrome.

There is a moral to this never-to-be-repeated evening of entertainment. The truth of the matter is that for many involved in the computer security business, the perspective is all wrong: one is simply too close to the subject to be able to make general decisions about how usable a product is. It is only by observing a user who needs to use a computer as a *tool* that one can begin to see the issues involved.

There is a large difference between the anti-virus software interface which one would need in order to carry out scheduled background scans and that which is used on an ordinary user's machine or a gateway PC. For the former, a highly configurable interface is required - no graphics, bells or whistles are needed (how long until someone releases a multimedia virus scanner?). On such a system, the information returned can be as technical or arcane as you like. However, a scanner on a user's machine needs to be simple, quick and convenient to use. Furthermore, the results need to be easy to interpret: all the user wants to know is whether or not a disk is infected. A scanner is not a video game, neither should it be confused with one. These different requirements should be remembered when choosing one. Something which is a personal favourite may well simply overwhelm a less technical colleague. Looks may not be important, but usability is.

NEWS

Good Times for None

The latest virus scare has had a file racing round the *Internet*, supposedly ready to infect a user's file on downloading. Many people have contacted *VB* for information: to this journal's knowledge, no such infected file exists.

Reports of the virus first surfaced at the beginning of December, with a message which appeared on the BBS *America On-Line* (*AOL*):

```
Here is some important information. Beware of
a file called Goodtimes. Happy Chanukah
everyone, and be careful out there. There is a
virus on America Online being sent by E-Mail.
If you get anything called "Good Times", DON'T
read it or download it. It is a virus that
will erase your hard drive. Forward this to
all your friends. It may help them a lot.
```

The US DOE's *CIAC* (*Computer Incident Advisory Capability*) immediately set up an investigation, tracing the message to two people: a user of *AOL*, and a university student. The perpetrators claim that the message was meant as a hoax.

The story has managed to spread as successfully as a 'good' virus, mainly due to the fact that many people who received a message with 'Good Times' in the header deleted it immediately, believing they had escaped infection, and then passed their story on. Another rumour related to this scare incident was that any file with the subject line 'xxx-1' was also infected.

There has been one confirmed report of a user who received a message with 'xxx-1' in the header: subsequently scanning his machine, he found a virus which could not have come in through his e-mail, but nevertheless reported it as related to the mail message incident. Unless one is using a mail program which allows either the automatic extraction of text into executable code or ANSI escape sequences, it is impossible for a mail message to 'infect' a computer.

All reports of the 'Good Times' virus received to date have been second-hand: *no-one* has had the 'virus' themselves. Until and unless this situation changes, *Virus Bulletin* will treat the story as based in fiction rather than fact, and advises readers to do likewise.

AOL spokesperson Pam McGraw said: 'We have looked into the reports and we haven't found any instances of it yet. We take these things seriously and we do look in to them.' She did not, however, rule out the existence of the virus.

If any user/reader believes that he has managed to capture this 'virus', or has factual information on the incident, he is urged to contact Pam McGraw of *America On-Line* on Tel. +1 703 556 3746; e-mail pammcgraw@aol.com, or *VB* on Tel. +44 1235 555139; e-mail virusbtn@vax.ox.ac.uk ▮

Virus Prevalence Table - November 1994		
Form	19	21.9%
Parity_Boot	12	13.8%
AntiExe.A	6	6.9%
JackRipper	5	5.8%
V-Sign	5	5.8%
AntiCMOS	4	4.6%
Tequila	3	3.5%
Stoned	3	3.5%
Exebug	2	2.3%
Halloween	2	2.3%
Monkey2	2	2.3%
Stoned.i	2	2.3%
Viresc	2	2.3%
1099	1	1.1%
3NOP	1	1.1%
AMSE	1	1.1%
Angelina	1	1.1%
Diskwasher	1	1.1%
Eddie-2100	1	1.1%
Helloween_1375	1	1.1%
Joshi	1	1.1%
Junkie	1	1.1%
Keypress-1216	1	1.1%
Keypress-1216-e	1	1.1%
Monkey	1	1.1%
Mutagen_1.10	1	1.1%
Natas	1	1.1%
Necros	1	1.1%
NoInt	1	1.1%
One_Half	1	1.1%
Stone-p	1	1.1%
Torj Lesbosex	1	1.1%
Yankee-2c	1	1.1%
Total	87	100%

The Team Expands

Virus Bulletin is pleased to announce the appointment of Roger Thompson, of *Thompson Network Softwar*e, USA, to its Advisory Board. Thompson has been an active researcher and developer in the anti-virus field for many years now, and brings with him much-valued experience.

As an introduction for readers who are not familiar with him, this month's Insight (see p.6) consists of an interview with Thompson, discussing his views on life, the universe, and anti-virus software ▮

IBM PC VIRUSES (UPDATE)

The following is a list of updates and amendments to the *Virus Bulletin Table of Known IBM PC Viruses* as of 19 December 1994. Each entry consists of the virus name, its aliases (if any) and the virus type. This is followed by a short description (if available) and a 24-byte hexadecimal search pattern to detect the presence of the virus with a disk utility or a dedicated scanner which contains a user-updatable pattern library.

Type Codes

- **C** Infects COM files
- **D** Infects DOS Boot Sector (logical sector 0 on disk)
- **E** Infects EXE files
- **L** Link virus
- **M** Infects Master Boot Sector (Track 0, Head 0, Sector 1)
- **N** Not memory-resident
- **P** Companion virus
- **R** Memory-resident after infection

Astra.927

CER: A 927-byte virus, which seems similar to the 976-byte variant reported in August 1992.

```
Astra.927          D88E C0BE 7900 03F5 8BFE B96F 018B DDFC AD33 8767 00AB E2F8
```

Barrotes.1194

CER: Detected with the Barrotes pattern.

Beer

CER: There are now two new variants of the Beer virus, one 2473 bytes long, which is detected with the pattern for Beer.3164, published in April 1993, and another, 3307 bytes long.

```
Beer.3307          FA90 80FC 3B75 03E9 1EFF 3D00 3D74 0F3D 023D 740A 80FC 5674
```

BigX.610

ER: The name of this 610-byte long virus is derived from the internal text string '[BigX]'.

```
BigX.610           80FC 9975 0333 C9CF 80FC 4E75 082E C606 7500 01EB 2080 FC4F
```

BootExe.394

ER: This virus infects EXE files by placing itself in the unused part of the file header, thus avoiding increasing the file size.

```
BootExe.394        A102 00BA 8000 2BC2 BE00 01BB 9201 8EC0 FCBF 0001 B900 02F3
```

Burger.542

CN: An uninteresting, overwriting virus - yet another rehash of Burger's published code. Detected with the Burger pattern.

BW

CN, CR, EN: The BW (Biological Warfare) virus development tool is a new addition to the virus author's arsenal. So far, the number of BW-generated viruses is only a fraction of those developed with VCL or PS-MPC, but we are seeing more samples every week. Most of the samples which have appeared recently have carried names like BWTEST_5.COM, indicating that the authors are experimenting with the program. The latest BW-generated viruses are: 525 (CR, encrypted), 556 (EN) and 756 (CN).

```
BW.756             B440 B9F1 0290 8D96 0601 CD21 32C0 E828 008D 96F3 03CD 215A
BW.556             B440 B92C 0290 8D96 0601 CD21 B800 4299 33C9 CD21 B440 B91C
```

Cascade

CR: Minor variants of this virus keep appearing. This month sees 1701.Y and 1701.Z, which are detected with the Cascade-1 pattern, as well as 1701.Yap.C, which requires a new one. Functionally, these variants are equivalent to a number of earlier viruses.

```
Cascade.1701.Yap.C  012E F687 2A01 0174 0 F8D B74D 01BC 8206 3134 3124 4C46 75F8
```

Danish_Tiny.163.C

CN: A minor variant, detected with the Tiny pattern.

Dark_Avenger.1800.M

CER: Detected with the DA-related pattern published last August. This variant has been modified somewhat, and the test strings from the original virus are not included.

Error_Inc.3 93

CN: An unremarkable, 393-byte virus.

```
Error_Inc.393      B989 01B4 40CD 21E8 5A 00 B43E CD21 B44F CD21 7203 E976 FFBF
```

Heja

CN: A simple, 511-byte virus which contains the string '(c) 1993 Heja/Adrar' in encrypted form.

```
Heja               B802 3DCD 218B D81E 5233 D2B0 00E8 43FF B905 000E 1FBA 0F00
```

Infector

CN: There are now two new viruses in this family, 469 and 875 bytes long.

```
Infector.469       B9D5 018B 1EFC 02B4 40CD 2172 2933 C933 D28B 1EFC 02B8 0042
Infector.875       A200 01A0 DD02 2EA2 0101 A0DE 022E A202 01B9 0001 BB00 002E
```

Intruder.1355

EN: Detected with the Intruder pattern.

Jerusalem.1808.Exciter

CER: Four new minor variants (A, B, C and D), containing the word 'Exciter'. Detected with the Jerusalem -1 search string. There are several other insignificant Jerusalem variants this month as well, 1808.Frere.J (detected with Jeru-1735), 1808.Sumsdos.AP and 1808.Sumsdos.AQ (detected with Jeru-1734, Jerusalem-1 and Jerusalem-US).

Junkie.B

CR: The decryption loop has been changed slightly, but otherwise this variant is practically identical to the original variant, now renamed Junkie.A. The MBR code is detected with the pattern published earlier.

```
Junkie.B            BB?? ??B9 F401 2681 37?? ??43 43E2 F7
```

Kode4.281

CN: Like the two other Kode4 variants reported last month, this virus contains the text '-=+ Kode4 +=-, The one and ONLY!'.

```
Kode4.281           803D E975 0D8B 4D01 582D 1901 3BC1 7502 EB2A 33C9 33D2 B 800
```

Leningrad_II

CR: There are three variants of the Leningrad virus - a 2000 byte unencrypted one, and two encrypted variants, 1499 and 2000 bytes. The encrypted 2000-byte variant has been named Leningrad_II.2000.B.

```
Leningrad_II (encr) 502E 8B36 0201 81C6 1901 B9?? ??B0 ??2E 3004 46E2 FA
Leningrad_II.2000.A 813E 8801 BEBE 7503 E988 000E 07B4 4ABB FFFF CD21 81EB 0301
```

Lockjaw.499

P: This uninteresting companion virus contains the text 'Good Night'.

```
Lockjaw.499         9C06 1E50 5352 3D00 4B75 03E8 0E00 5A5B 581F 079D 2EFF 2EF3
```

Milan.Demon.270

CN: This is a minor variant of an old 272-byte virus. It contains the texts 'Demonhyak Viri X.X (c) by Cracker Jack 1991 (IVRL)' and 'Error eating drive C:'. The virus overwrites the files it infects and is not likely to spread at all. It is detected with the 'Demon' pattern, which should be renamed to Milan.Demon.

Nygu s.278

CN: Similar to the three variants reported in December '93, but 278 bytes long, and contains the text '(c)Nygus v1.1'.

```
Nygus.278           B440 E83C 00B4 3ECD 21B4 4FBA 2B02 E830 0072 03E9 7BFF B905
```

PHX.1289

CER: Not fully analysed, but seemingly a more advanced version of the variant reported in last month.

```
PHX.1289            80FC 4B74 7D3D 023D 7443 3D79 B974 EA80 FC40 74DA 80FC 4E74
```

Pixel.8 52.B

CN: A minor variant, detected with the Amstrad and Pixel-936 patterns. There are two other Pixel variants this month. Both contain destructive code which overwrites disks.

```
Pixel.1577          B916 04BE 0E01 8BFE AC32 C4AA E2FA 33FF 8E06 2C00 33C0 B590
Pixel.1686          B916 04BE 0E01 8BFE FCAC 32C4 AAE2 FA33 FF8E 062C 0033 C0B5
```

PS-MPC:

CN, CR: Two new PS-MPC viruses this month: Dangler (CN, 298) and Happy_Day (CR, 475).

Tai-Pan.666

ER: This virus has also been reported as 'Doom II Death', but that name is derived from a string it contains: 'Illegal DOOM II signature Your version of DOOM2.EXE matches the illegal RAZOR release of DOOM2 Say bye-bye HD The programmer of DOOM II DEATH is in no way affiliated with ID software. ID software is in no way affiliated with DOOM II DEATH.'

```
Tai-Pan.666         1658 0306 9802 A3AF 00A1 9602 A3AD 0016 582D 1000 8EC0 8ED8
```

Teraz.4004

CER: This is a large, complex virus, probably of Polish origin - at least it is related to a 2717-byte Polish virus, and contains the text: 'TERAZ POLSKA v2.0 produced by NoName All rights reserved (c) 93.12.22 POLAND.'

```
Teraz.4004          3D35 FF74 2680 FC4E 7411 80FC 4F74 0C80 FC11 740F 80FC 1274
```

VCL

CN: The new VCL viruses this month are 634, Anston (2126 bytes) and Mindless.423.B (overwriting).

Vienna.435

CN: There are now nine new variants, which have been named 435.C-K. Vienna.435.C is detected with the Vienna-6 pattern, but the rest require new search strings.

```
Vienna.435.D        8E1E 2C00 AC3C 3B74 0B90 3C00 9074 03AA EBF2 2BF6 1F89 768A
Vienna.435.E        8E1E 2C00 AC3C 3B74 0F90 903C 0090 9074 0590 AA90 EBEE 2BF6
Vienna.435.F        8E1E 2C00 AC3C 0074 073C 3B74 05AA EBF4 33F6 1F89 768A 807D
Vienna.435.G        8E1E 2C00 AC3C 0075 03EB 0890 3C3B 7409 AAEB F157 33FF 8BF7
Vienna.435.H        8E1E 2C00 AC3C 0075 03EB 0B90 3C3B 7503 EB0A 90AA EBEE 5733
Vienna.435.I        8E1E 2C00 AC3C 0075 03EB 0B90 3C3B 7503 EB0A 90AA EBEE 572B
Vienna.435.J        8E1E 2C00 AC3C 0075 03EB 1490 3C3B 7503 EB0F 90AA EBEE 572B
Vienna.435.K        8E1E 2C00 AC3C 0075 03EB 1B90 3C3B 7503 EB16 90AA EBEE 572B
```

Vienna.Violator.680.B

CN: Detected with the Violator pattern.

Vienna.Violator.821.B

CN: An utterly insignificant minor variant, detected with the following pattern.

```
Violator.821.B      ACB9 0080 F2AE B904 00AC AE75 EDE2 FA5E 0789 BC79 008B FE81
```

Vienna.565

CN: Detected with the Vienna-6 pattern.

Vienna.1006

CN: Detected with the Vienna-4 pattern.

Wordswap.1503.B

CER: A minor variant of a virus reported in November 1991. No search pattern is possible.

WVP.352

CR: Similar to the variant described in July '93, but somewhat shorter.

```
WVP.352             3DD0 D075 01CF 3D00 4B74 1480 FC43 740F 80FC 5674 0A80 FC3D
```

INSIGHT

Thompson: Wizard from Oz

Megan Palfrey

Roger Thompson, author of *Doctor*, has been around computers since the days when having 64K of RAM meant an enormous computer. Indeed, the ICL 1904-A, which was the first computer he ever programmed, was such a monster that, at the time (the early 1970s), it was the largest computer in the Southern Hemisphere.

Throughout these early years, he was employed as a mainframe systems programmer, finally starting his own software business in 1980: 'We called it *Microsoft*. Eventually, I found out about this other company in the USA which had "stolen" my name - unfortunately, at least two years before I started - so I traded the Australian rights to the name for a new C Compiler! As I said, these were early days.'

Crocodile Farms and Pizza

Thompson remembers the difficulties of writing programs in those 'micro' days: 'Everything, from accounting applications to ISAM routines, sorts, screen generators and even communications protocols, had to be written from scratch. My programmers worked round the clock, drinking coffee and eating pizza - one of the great productivity advances of that period was the advent of the *Dial-A-Pizza* delivery service! We did everything, from creating a point of sale [cash register] system, to writing a growth-tracking program for a crocodile farm in Papua New Guinea.'

He went from these diverse and exotic projects to writing database applications for State and Federal Government departments: it was at about that time that he first started hearing about computer viruses.

'I paid no attention to them,' he said, 'I always thought that they were a press 'beat-up', or someone else's problem, or that, if I ever did encounter one, I would instinctively know what to do because, hey, I was a Tech Guru.'

Infection

In 1988, Thompson's dreams were rudely shattered by what he described as 'worse than a virus' - a false positive. This spurred him into realizing that viruses had the potential to be a serious problem.

'I kept thinking,' he said, 'how easy it would be if only I had a reliable checksum of the boot area and of each executable. So, I contacted a friend, Jack Kenyon, who had developed a file management program like *XTree*. His was written in Turbo Pascal, which gave it a nice clean interface, and easy access to DOS and BIOS system calls. We realized that if his program was just to checksum each program found, it would be a reasonable anti-virus program. I suggested we call it *Virus Buster*, and we formed *Leprechaun Software* to develop and sell the product.'

While Kenyon was developing the checksummer, his partner was disassembling Stoned, which had just appeared in Australia, and writing the company's first cure program, *Doctor*. Although effective, it was written in Assembler, and different from *Virus Buster*. Thompson soon turned *Doctor* over to Kenyon, so that it could be developed along the same lines as the checksummer, and began concentrating in earnest on virus disassembly.

Across the Pond

In 1991, *Leprechaun* branched out into the USA, to try and find a new market for *Virus Buster*. After Thompson moved there, it became apparent that his views had diverged sufficiently from those of Kenyon's to make continuing as partners difficult; thus, *Thompson Network Software* came into being.

> *"your products need to run on any platform, and need to cause a minimum of problems"*

'Sadly, splitting a company is about as difficult as getting a divorce, and almost as acrimonious,' reflected Thompson. 'In fact, the paperwork is still with the attorneys, and being redrawn again. I expect that it will be concluded by about Christmas, and I wish him well in his future endeavours.'

A New Look

The product *Doctor*, now being developed by *Thompson Network Software*, is based on the original *Virus Buster* but has a new scan engine written in C. This is different from the *Leprechaun* version, which Kenyon has also recently rewritten and continues to develop.

Thompson wants a product which is all things to all people, something a 'techie' will enjoy using just as much as an end-user: 'Reviewers look at a product for its technical ability, but customers want something different. I know many techs whose personal favourite is not released company-wide because it is too complex and powerful.

'For their users, technical staff want something easy to update, so that they can get it to their users with no extra work; pretty much automatic, so end-users will not have to think; lightning fast, so users will not complain; and something which causes neither compatibility problems nor false alarms, and misses nothing.' He hasn't worked out whether this is possible, but plans to keep trying!

Roger Thompson (right), 'tech guru', dispensing words of wisdom at the *20th Annual CSI* conference.

Holistic Heuristics?

Despite '*Symantec's* recent acquisition frenzy', and the plethora of products available on the market, Thompson does not see the number of anti-virus product developers decreasing greatly: 'There are two reasons for this, the first of which is that the very nature of our business requires us to stay close to our customers. Everybody releases a version at least monthly, and generally, we are all able to be very responsive. This breeds intense customer loyalty. Secondly, most developers are small and thus have extremely low overheads. Unless these small vendors wear themselves out disassembling the legions of new viruses, I see no reason for any to drop out.'

This will lead, Thompson feels, to more consistency within the top products: 'Already, the scanner shootouts show that the top fifteen score within five or six percent of each other. Most now incorporate a checksummer, a behaviour blocker, and a virus-specific scanner. Within twelve months, I feel sure that everyone will have a heuristic module.'

Heuristics are a good idea, in Thompson's book: 'It is pretty darn handy to have your program look at a new disk or file and say something like "I don't see a known virus here, but I can see the following suspicious pieces of code", prior to using a disassembler. Many end users, however, cannot make proper use of that sort of information, so in the overall scheme of things, heuristics becomes just another layer, like checksumming or virus-specific scanning.'

Thompson Network Software takes the view that the real need in anti-virus protection is neither server- nor workstation-based, but enterprise-oriented, requiring a mix of server and workstation products which are centrally as well as locally configurable and updatable.

'Your products need to run on any platform, and need to cause a minimum of problems. Back that up with some really good clean-up products, and strong technical support. That's where I think it is all headed.'

Virus Writers

A recent trend has seen virus authors writing viruses which target the products of particular companies. This is not a major concern, said Thompson: 'We all know they can write a virus that can defeat one product for a while, but only for a while. I think the older virus writers will eventually get sick of it, and find something more interesting to do.'

Sadly, he does not see this as an end to viruses: whence comes one young virus writer, hence come many more to pick up the baton. New writers, in his view, will develop typically trivial first attempts. These, with little more than nuisance value, will form the majority of incidents, but the experienced writers will go on to develop more advanced viruses: stealth, multi-partite, polymorphic, retro.

Thompson believes that it is these more complex viruses which will have the most catastrophic effects; causing serious downtime, loss of money and sometimes loss of irreplaceable data. Virus authors will be spurred on to new 'creative' efforts by the temptation of new operating systems such as *Windows NT*: 'Already I have seen some articles in hacker magazines pointing out that it is almost 1986 again.'

The Flip Side

Like many computer people, Thompson has an 'other' life, as a clubbing guitarist in his wife Kate's band. They have been playing the Atlanta circuit for the past few years, and have released a CD of their own music. Despite his conviction that Kate will be a success, his principal focus will remain *Thompson Network Software.*

'It would sure be fun to play some big stadiums… But I'm a programmer, pure and simple. I am not writing too much sellable code at the moment, but that's by design. I disassemble my share of viruses, and still produce some nifty in-house tools, but I just don't have time to polish them for commercial sale. I have three excellent programmers: it makes more sense to let them do the commercial stuff - they can concentrate on it, while I get pulled all over the place. It also allows me to make sure our expertise is spread a bit.'

Thompson plans to expand his endeavours in the USA: from a personal point of view, he would also like to be with the rest of his family in Australia again, but with a population difference of over 200 million, opportunities for software and for music are greater in the US: 'But one should never kid oneself. The very size of this market presents enormous difficulties as well as enormous opportunities.

'It was difficult to get started here: by the time we arrived, the big guys had already become interested and had started to crank up their marketing engines. However, I know that if you are prepared to commit every resource you have, be patient and take a long term view, keep your nerve, and take some reasonable chances from time to time, it's easy,' he laughed. What lies ahead? 'We are very confident that we have a strong foundation for the future,' said Thompson. Well, time will tell.

VIRUS ANALYSIS

Hiding in EMS - A Creative Crisis?

Eugene Kaspersky
KAMI Associates

Every year brings new developments in virus-writing to keep anti-virus vendors on their toes. Since the development of the Brain virus, the number of infection techniques has grown, and virus authors have added stealth, encryption and polymorphism to their armoury.

The result of this is that computer specialists have had no option but to become living encyclopædias of computer programming. They must know operating systems such as *MS-DOS*, *MS-Windows*, and *OS/2* inside out, in order to be combat a rapidly shifting threat.

They must also be familiar with the modes of different processors (to be able to stop the spread of protected-mode viruses such as PMBS), and about the different hardware components which collectively form the PC.

Knowledge of other computing techniques is useful: take for example the Cruncher virus, which compresses itself using the DIET algorithm. The latest addition to the 'bag of tricks' used by researchers is brought by the Emma virus, which places its code in expanded memory.

EMS Virus Installation

This virus is a 427-byte long, memory-resident, appending parasitic COM file infector. When an infected file is executed, the virus receives control through a JMP instruction placed at the beginning of the infected file.

Once loaded, Emma tests system memory for an already active copy of itself by using an identification word at the address 0024:0000h. If the address contains the word 2E9Ch, the virus passes control to the host program.

To install itself into system memory, the virus makes several calls to the EMS driver. First, it checks that the EMS driver is in fact present. It calls the Get_Vector DOS function to obtain the address for Int 67h (the LIM EMS interface), and checks the memory area to which that vector points. Once that address has been obtained, Emma then compares the data there, at offset 000Ah, with its own internal string EMMXXXX0. This will determine whether the EMS driver is loaded in the system. If no driver is found, the virus does not become memory-resident, and processing is returned to the host file. This therefore provides an easy way to remove the virus from memory on an infected computer: if the EMM driver is removed from CONFIG.SYS and the computer rebooted, the virus will no longer function.

If an EMS driver is found, the virus then obtains the number of unallocated pages in EMS, using the function Get_Number_of_Pages (Int 67h, AH=42h). Control is returned immediately to the host program if no free pages are found.

If, however, there are free pages, the virus finds the EMS frame segment address with the Get_Page_Frame_Segment function (Int 67h, AH=41h), stores that address, allocates one EMS page with the Get_Handle_and_Allocate_Memory function (Int 67h, AH=43h), and makes it available (maps that page into standard 640K memory) with the Map_Memory function (Int 67h, AH=44h). Then the virus copies itself into that frame (i.e. it copies its code into EMS memory) and unmaps the page.

"the first block, which is placed in EMS memory, contains the complete virus body, and cannot be accessed without special calls to the EMS driver"

Once this is done, the virus code is stored in EMS memory. The remainder of the installation routine copies the virus' Int 21h handler code (42h bytes) into the Interrupt Vector Table at address 0024:0000h, hooks the Int 21h vector, and returns control to the host program. The address of the virus' Int 21h handler is the same as address of the virus' ID-word, i.e. the first two bytes of the Int 21h handler's code.

Int 21h Handler

When the installation routine is complete, the virus has installed its code into two areas of system memory. The first block, which is placed in EMS memory, contains the complete virus body, and cannot be accessed without special calls to the EMS driver. The remainder, the Int 21h handler's code, is inserted into the Interrupt Vector Table, which is part of standard 640K memory.

The reason for this division is quite simple: any program which saves its code and data in expanded memory blocks must keep some of its code in conventional memory, or in other memory blocks below the 640K limit, in order to make the EMS code/data available via calls to the expanded memory manager.

As Emma's Int 21h handler installs the greater part of its code in EMS memory, it must also put a 'stub' of its code into the Interrupt Vector Table. The presence of this stub allows virus scanners to detect the virus without scanning EMS memory.

The block of code which is installed in the Interrupt Vector Table is very short, and contains only three calls. The first of these is the Map_Memory function, which makes the virus' EMS block in the frame segment address discussed above available. Next comes a far call to that frame, and finally, the last call, which will unmap the virus' own EMS block.

File Infection

When the main code of the virus takes control of a file, the virus first checks whether the call is the DOS function Load_and_Execute (Int 21h AX=4B00h). On such calls, the virus allows the original Int 21h call (i.e. the Load_and_Execute function) to complete and only then infects the executed file.

The virus opens the file, reads its header, and makes a check to avoid multiple infection. This is carried out by examining the first instruction of the file. If it is a JMP instruction, the offset to which processing jumps is calculated. Should this be 430 bytes from the end of the host file, it is assumed that the file is already infected, and control is returned to the calling function.

If the file passes this test, the virus checks the file header for the EXE stamp (the word MZ). If that stamp is found (that is, if the file is in EXE format) the virus aborts its infection routine. If it is not (i.e. it is a COM file), the virus writes its body at the end of the file, adds a JMP VIRUS instruction at the beginning of the host file, and returns control to the calling code.

The routine which carries out the infection process is extremely simple. No attempt is made to preserve the infected file's time and date stamp, and no check is made of the file attributes. Finally, Int 24h is not hooked, so any calls to the critical error handler are passed though unhindered.

As a corollary, protecting files from infection is not at all difficult: it is necessary simply to set their attributes to read-only. Detecting the virus on execution of COM files from write-protected disks is also trivial, as DOS will display the standard error message:

```
Write protect error writing drive ? Abort,
Retry, Fail?
```

Conclusions

We are now seeing the advent of viruses which can insinuate themselves into the EMS system area, hiding there. The first example of this type of virus appeared only this year, despite the fact that there is nothing technically new or complex involved - indeed, this virus could conceivably have been written some time ago. Why then only now? Could it be that the virus writers are having a 'creativity crisis', a crisis of ideas?

One of the best collections of the ideas used by virus writers is in *Virus Bulletin*: a brief scan through the pages of this journal reveals a wealth of information - polymorphic engines and virus constructors; boot and multipartite viruses; parasitic COM, EXE, SYS, OBJ, and even Source Code viruses; viruses which work in protected mode; memory-resident viruses which copy themselves into the Interrupt Vector table, into DOS system data, into conventional and high memory... there are any number of ideas to be found. And now, to add to the 'catalogue', there are viruses which can copy themselves into expanded memory blocks, causing no decrease in low memory.

Is there still more to come? Is it still possible for virus authors to invent yet more techniques to affect *MS-DOS*? The answers to such questions lie in pure conjecture: we can only hope that the flood of ideas has indeed started to recede, and that the Emma virus turns out to be the last drop from the shower.

Of course, all this applies only to *MS-DOS* viruses. Technology goes on apace, and there are still virgin lands waiting to be conquered: *OS/2*, *Novell NetWare*, *Windows95*...

Emma

Aliases:	None known.
Type:	Memory-resident parasitic COM file infector.
Infection:	Any file which does not begin with 'MZ' executed via Int 21h 4B00h.
Self-recognition in Files:	Checks file header, see text for further details.
Self-recognition in Memory:	Compares word at offset 0024:0000h (Interrupt Table) with 2E9Ch. These are the first two bytes of the virus' Int 21h handler.

Hex Pattern:	In files:
	`602E 8B1E 0101 81C3 0301 E80A`
	`008B FE8D 7703 A4A5 61FF E653`
	The following pattern can also be used to detect the presence of the virus in low memory:
	`9C2E 803E 3B00 0075 3390 9060`
	`B800 4433 DB2E 8B16 3900 CD67`

Intercepts:	Int 67h for location and allocation of free page(s) in EMS driver (the DOS LIM EMS driver interface) and Int 21h subfunction 4B00h (Load_and_Execute) for infection.
Trigger:	None.
Removal:	Under clean system conditions, identify and replace infected files.

TUTORIAL

Virus Infection Techniques: Part 3

This article is the third instalment in a series which examines the various infection techniques employed by virus writers. The series examines various infection techniques employed by virus writers, and is intended to form a source of reference for anyone involved in virus prevention.

Three more infection strategies are considered this month: link viruses, cavity viruses, and the method employed by Commander Bomber.

Link Viruses

Link viruses (also known as 'file system' or 'cluster' viruses) have caused sufficient misunderstanding already to have merited an in-depth discussion of their infection technique in *Virus Bulletin* (see April 1994, pp.12-13). However, to complete this series, a brief overview of their operation is given here.

Unlike most viruses, link viruses do not alter the code of the host program; rather, they alter the pointer to the start of the host file, changing it to point to their own code. In order to understand link viruses fully, it is necessary to consider how the operating system accesses files under DOS.

Each DOS partition has four distinct parts: the boot area, the File Allocation Table (FAT), the Root directory, and the file area. The boot area contains a small program for loading the operating system and a table known as the BIOS Parameter Block (BPB), which lists details of the disk's structure (total number of sectors, number of bytes per sector, etc). When the operating system is loaded, it uses information stored in the BPB to locate the other critical areas of the disk.

The directory tree is built up from entries in the Root directory. Each Root directory entry contains a file name and extension, the file's attributes, its starting cluster number, its time and date stamp, and the file size. When a directory entry is accessed, DOS uses the starting cluster number to locate the remainder of the file. This is carried out using information stored in the FAT, which consists of a linked list relating each cluster used by a file to the next.

It therefore follows that if the starting cluster for any file is altered, the operating system will associate the new cluster (and any others linked to it by the FAT) with the file. When a link virus infects a file, it must alter only the file's starting cluster to load itself instead of the host file. Once the virus has become memory-resident, it loads and executes the host program from the disk.

This infection algorithm has distinct advantages over appending file infection: firstly, the virus can infect every executable file on the disk using only one copy of its code. Secondly, infection is very quick, as only minor changes need to be made to the disk.

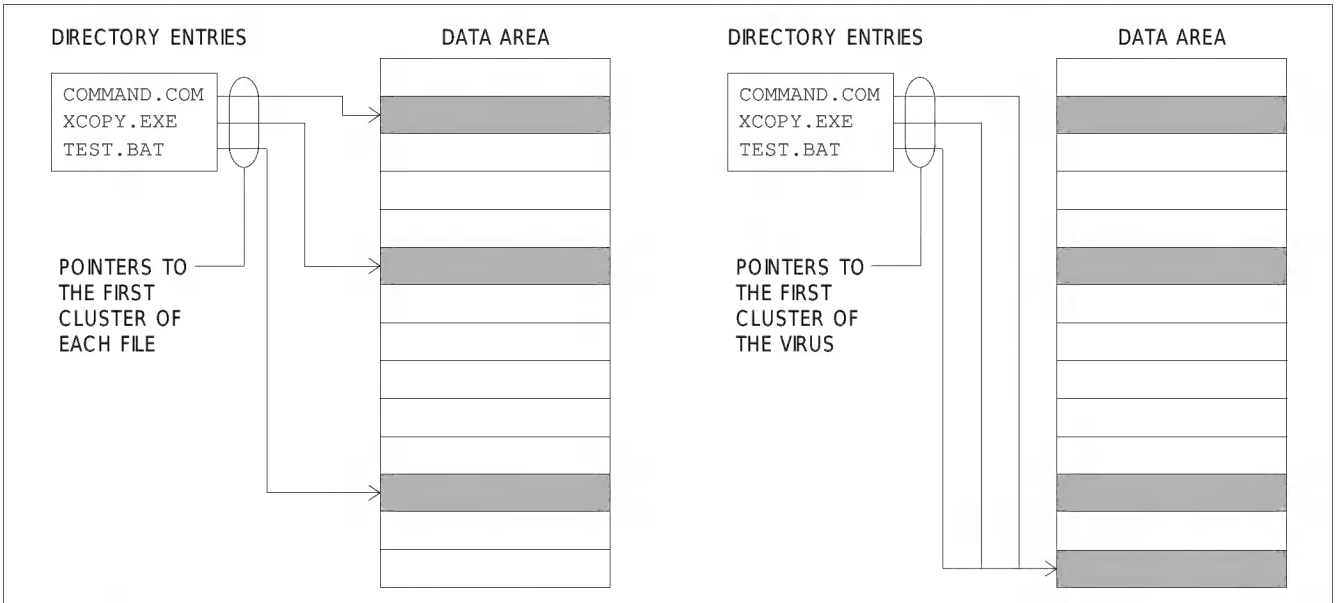


Figure 1: Link viruses operate by altering the pointer to the first cluster of an infected file. In the diagram above, the left hand side represents an uninfected file system. Each Root directory entry points to a different starting cluster. The right part of the diagram shows an infected system. Here, every directory entry points to the same cluster, which contains the virus code.

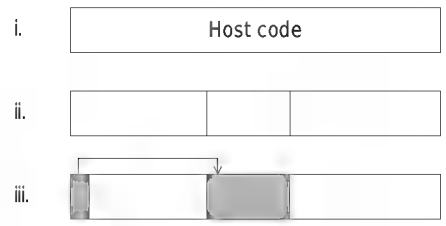


Figure 2: A cavity virus in operation. i. An uninfected host file. ii. The virus searches for an area within the host file filled with a constant value. iii. The virus overwrites the 'hole' in the host file, changing the entry point of the file so that control is passed to the virus code upon execution.

It is not difficult to detect a link virus under clean system conditions, as all infected files will be seen to share the same starting cluster. Similarly, disinfection is easy, as the host files have not been altered: only their starting sector has changed. To disinfect an affected file system, anti-virus software need only restore the pointer to the first cluster of a file; no changes to the EXE code are necessary.

Cavity Viruses

A specific instance of the cavity virus, the EXE Header virus, has already been discussed in Virus Infection Techniques Part 2 (see *VB*, December 1994). This technique allows infection of both COM and EXE files, without any increase in file length.

The underlying infection algorithm of this type of virus is a simple extension of appending parasitic file infectors. However, instead of adding its code to the end of an infected file, a cavity virus searches for an area of the file filled with a constant value (usually zeros). If this area is large enough, the virus stores that constant value and inserts its code into the 'cavity' within the file, altering either the EXE header, or the start of a COM file, to pass control to the virus code. When an infected file is executed, the virus code repairs the memory image of the host file by overwriting its own code with the constant value it contained originally.

Cavity viruses present no unusual problems in terms of detection: although file length is unchanged, the entry point of infected files is altered to point to the virus code. Thus, both scanners and checksummers can detect the presence of the virus code with little difficulty. The only complication is that the offset of the virus code within the host file varies depends on the location of the cavity, making detection of polymorphic code harder, as no fixed offset is available.

In terms of disinfection, the anti-virus software must rebuild the executable file using information stored in the virus. This may cause problems with certain generic techniques, in particular those which rely on storing the beginning and end of the original uninfected host file.

Commander Bomber

When the Commander Bomber virus was first discovered, a small tremor passed through the anti-virus community. For the first time, a virus which threatened to cause severe detection problems for anti-virus software developers had been created.

Two years later, it appears that Commander Bomber did not represent the start of a new era in virus writing. However, the fear was not wholly unjustified: the implicit threat posed by the infection technique used by Commander Bomber is just as real today as it was then.

Essentially, Commander Bomber is the fusion of two ideas: the insertion of virus code at a random offset within an infected file, and an unusual and complex way of passing control to the added virus code. Because its code is not located at any fixed offset within the host file, this virus is particularly difficult to detect.

Locating the start of the virus code by simply following the first instruction of an infected file is not possible, due to the manner in which control is passed to the virus code. Fortunately, the body of the virus is not encrypted, so a 'brute force' scan of an entire file will detect it. However, the virus would be harder to detect if a polymorphic engine were added to the body of the virus code: it would then be necessary for a scanner to be able to identify the polymorphic code scattered throughout the file.

Since Commander Bomber was first written, scanners have improved, and can now cope with most of the problems posed by this infection technique. Changes made to an infected file will be detected by integrity checkers; thus, although the threat is not negligible, there is now little or no reason for scanner manufacturers to have to search an entire file in order to detect the virus' presence.

Disinfection of infected files is difficult: the virus code is scattered throughout the host file, and the anti-virus software must either reverse-engineer the changes, or allow the virus code to run and repair the host file itself. Commander Bomber further complicates the situation with its tendency to damage infected files, destroying information.

Conclusions

The more esoteric infection algorithms pose problems to anti-virus software manufacturers, in terms of both detection and removal. However, due to their rarity, users have yet to experience the practical difficulties associated with these techniques. Doubtless vendors will address these issues when such viruses are encountered 'in the wild'.

Next month, the final instalment in this series, 'Virus Infection Techniques', will examine viruses which have unusual methods of infection, including Object file infectors and source code viruses.

FEATURE

The Whole Story

Dr Keith Jackson

The plethora of anti-virus products currently on the market presents a bewildering array to the naïve user. Manufacturers can sometimes add to the confusion with impenetrable jargon and silly slogans, making what are often spurious claims about their product's unique anti-virus features.

This article represents an attempt to penetrate the advertising jargon and get through to the actual capabilities of each method of protection, assisting would-be purchasers to make a reasoned selection.

Scanners

The most widely-used protection against computer viruses is the scanner, which is the primary line of defence employed by most anti-virus products. A scanner inspects an executable file to determine whether it is infected, examining file contents for patterns or code sequences specific to a virus.

Some products incorporate 'Artificial Intelligence' techniques (whatever that means!), which can supposedly learn about new viruses, and extract signatures automatically. I have yet to encounter one that does what it purports to do: if it *is* possible to extract signatures automatically using only software, why don't vendors of scanner programs use the technique themselves? The reason is that they don't work.

Many scanners include heuristic features. This is scientific-speak for guessing: i.e. the product will attempt to make intelligent guesses as to the presence (or otherwise) of a virus without being able to state with certainty which virus is actually involved.

The exact type of heuristic feature is more important than the fact that such features are being used *per se*. Therefore, always ask the vendor of an anti-virus product claiming to incorporate heuristic features what precisely it does.

Integrity Checks

The next most common method of virus detection relies on ensuring that executable files have not been altered, i.e. on checking their 'integrity'. Various names, such as 'checksumming', 'signature checking', and 'file verification', are used to describe the process. When an integrity checker is first installed, a checksum is made for each file to be protected. If a file changes in any way, the checksum of the file will alter.

When used in the context of checking file integrity, the word 'checksum' is probably more prone to hype and jargon than almost any other word used by anti-virus vendors. It is also called a 'hash-value', an 'integrity check', or a 'signature' (wonderfully confusing, given that scanners look for virus signatures). These all mean the same thing - a value is calculated which is a function of a particular file's content.

In most circumstances, an integrity check is capable only of detecting that changes have happened in a particular file. Nothing is known about where or why the alterations were made. A virus, a program that modifies its own executable file, or another security product (see Inoculation below) can all cause a checksum to change. This means that integrity checks are prone to reporting problems when no sinister reason exists. However, they *can* detect changes caused by unknown viruses, which scanners cannot.

Monitor Programs and Behaviour Blockers

A monitor program captures interrupt vectors, overseeing the operation of a computer with the aim of detecting the presence of a virus before it does any damage. Most monitor programs operate as memory-resident software, although some use special-purpose hardware (see below).

Unless a monitor program has an exceedingly large overhead, it should be able to carry out its tasks transparently. Avoid those that cannot. An addition to this technique is often to launch a 'goat' executable, in the hope that any virus resident in the system will infect it. This approach reduces the problem of false positives.

Monitor programs often have difficulty in discriminating between 'normal' activity (e.g. an executable file, which alters it own contents) and virus activity. They can scan files, verify checksums, detect unauthorised file writes, etc: the list is almost endless. If you are considering purchasing such a program, ask the vendor to provide a clear, unambiguous list of the functions monitored, and the circumstances in which the program will activate.

Like a monitor program, a behaviour blocker usually comes as memory-resident software, or as one function in a piece of anti-virus hardware. Its purpose is to detect any unusual behaviour, e.g. changing the contents of the CMOS RAM, or formatting the hard disk. Most behaviour blockers ask for confirmation that an action is required before allowing it to proceed. As such, their nuisance rating is often high, and they can cause a high level of false alarms. It is left to user to decide whether an action is legitimate - a decision which requires an intimate knowledge of a program's functionality.

Disinfection

Removing a virus from an infected file is known as disinfection; it usually provides less than ideal results. Many viruses destroy parts of an infected file, or insinuate themselves so subtly into a file that perfect restoration of an infected file is impossible. As

very few reviews of disinfection capability have been carried out, the true effectiveness of such software is open to question.

Many products claim to offer 'generic disinfection': i.e. they have the ability to rebuild a file through information the product has stored on the file's contents. Short of keeping a copy of the entire file, this would be possible only where the virus concerned was obliging enough to infect in the same manner as most other viruses. Wishful thinking perchance?

"Unless backups are unavailable there is no sensible alternative to replacing affected files"

Another generic disinfection technique relies upon the fact that most viruses restore the host file in memory and execute it. Therefore, it is theoretically possible for an anti-virus product to execute the virus code under its own control, and save the repaired memory-image of the file. Unfortunately this relies rather too heavily on the virus author's competence for comfort!

Ensure that you have clean and trusted backups/master disks for any files which could have become infected or damaged, as there is no truly reliable alternative to replacing such items as these.

Inoculation

Some products 'inoculate' a file by adding small amounts of executable code to the file, which is executed before the file's original content and which inspects the file for viruses and/or alterations before execution is allowed to proceed.

In adding itself to an extant executable file, inoculation acts in exactly the same way as a virus. Many viruses introduce changes to their host files which prevent the host software from executing; introduction of extra code by inoculation can have the same problem. It is difficult enough to maintain executable files in their original condition without complicating matters by using methods which alter them purposely.

Hardware Write-protection

Write-protection (when performed in hardware) is exceedingly significant to anti-virus defence. Several products offer software versions of write-protection: where implemented, it can be circumvented by a virus. As part of the hardware, however, that cannot happen, no matter how devious the virus software. Write-protection is the *only* guaranteed way to ensure that nothing has been written to a particular disk.

Special purpose hardware with anti-virus features takes control of the PC before the operating system is loaded, even before booting - this is invaluable in protection against boot sector viruses. Such hardware cards may offer almost any facility described in this article, but are particularly suitable when providing a monitor program and/or a behaviour blocker, write-protection for hard disks, memory protection, and detection of changes to interrupt vectors.

Backups and Encryption

Adequate, up-to-date backups are the single most important defence against virus attack. If an executable program is corrupted, replace it from a known good backup. Do not gamble with disinfection.

Any program which insists on removal of write-protection from its master disk, and will not let you work from a copied disk, is copy-protected. Most companies who develop such software do not like this terminology, using other words to describe it. The test to discover whether or not a product is copy-protected is simple: can you take as many copies as you want, and operate *fully* using any of the copies? If not, it is in some way copy-protected, and should be avoided.

Many products also offer encryption, either on a file-by-file basis or as an encryption facility for an entire disk. Encryption is applied before information is written to disk, and decryption before it is made available for use.

Although undoubtedly a powerful technique when used to prevent access to data and/or executable files, encryption should be used with great care where viruses are concerned. Many viruses fail to detect the presence of encryption and attempt to make disk alterations anyway. This could be little short of catastrophic unless adequate backups are available.

Disk Validation Software and Access Control

An important category of anti-virus software is disk validation programs. Such software prevents the use of 'unauthorised' diskettes on protected machines. In order to 'validate' an incoming diskette, it must first be scanned for viruses. This allows the IT department to have at least some guarantee that incoming media is checked before use.

It is possible to take this one step further and install access control software. Although it is received wisdom that access control cannot stop viruses (but merely slows them down), restrictions on the import of executable code via removable media will naturally help prevent use of unauthorised or pirated software.

Conclusions

This is a description of the methods of defence against viruses which are currently available - I would need clear, persuasive evidence that any new technique was genuinely novel before I could be persuaded to add to my list. However, new tactics will no doubt be developed, and new anti-virus methods may be needed at some future date.

I certainly advise the use of multiple layers of anti-virus defence, and advocate the use of more than one scanner. However, they are not the only solution: the best protection is provided by a multi-layer approach.

COMPARATIVE REVIEW

The 1995 Scanner Top Ten

Looking at the introduction to *VB's* July 1994 review of DOS-based virus scanners, one can see that the last six months have been 'business as usual' for manufacturers.

Since then, some names have disappeared, and new ones have been added. The number of viruses has continued to climb, and polymorphic virus detection has been made more urgent by the encounter of two highly polymorphic viruses in the wild: Smeg.Pathogen and Smeg.Queeg. Although neither is widespread, they highlight the need for vendors to continue to improve - the battle with the virus authors is no nearer to completion than it was then.

This review expands on the theme of polymorphic virus detection, and has a distinctly 'in the wild' flavour. The Boot Sector, Polymorphic and In the Wild test-sets have all been extensively revamped, making these the toughest benchmark tests which products have ever had to undergo.

Testing Protocol

Products were put through their paces against four test-sets: 'In the Wild', with 126 samples of file infectors known to be causing a problem in the 'real world'; a Boot Sector collection, consisting of 11 boot sector viruses frequently encountered in the wild; the 'Standard' test-set, consisting of 230 file infectors, and finally, the 'Polymorphic' test-set, which contains a mammoth 4796 infected files. For full details of the test-sets, see the table at the end of the article.

Disk scanning speed on an uninfected *Bernoulli 90*, an uninfected diskette and an infected diskette was measured for each product. All tests were performed on a *Compaq Deskpro 386/20e*, with a 112 MByte hard disk and 4 Mbytes of memory.

Product speeds are given in kilobytes per second. This represents the times taken to scan an uninfected *Bernoulli 90* containing 1430 executables file spread across 40 directories and occupying 76,049,880 bytes. The test diskettes used were both 1.44 Mbyte 3.5-inch floppy disks. The clean diskette contained 18 EXE, 19 COM and 12 SYS files, occupying 1,433,709 bytes. The infected diskette contained 100 EXE and 58 COM files, all infected with either Groove or Coffeeshop, and occupying 1,413,319 bytes.

All tests were carried out using each product's default settings, except to turn off any audible alarms when viruses were discovered, or to allow the product to proceed without user interaction during detection tests. Overall positioning of products is made by considering only the total number of viruses detected. The reader is referred to the appropriate edition of *Virus Bulletin* for full details of any product.

AntiVirus+ v4.20.29

In the Wild	90.5%
Boot Sectors	63.6%
Standard	97.4%
Polymorphic	12.6%

An uninspiring set of results from *Iris Software*, especially against the boot sector and polymorphic test-sets.

Anyware AntiVirus v2.15

In the Wild	82.5%
Boot Sector	72.7%
Standard	98.3%
Polymorphic	10.4%

A new contender in the *Virus Bulletin* comparative review, this Spanish product ships in a box covered with awards from various magazines. Unfortunately, the scanner contained within it does not seem to be able to deliver the goods, especially when pitted against the very tough Polymorphic test-set.

Avast! Version v7.00

In the Wild	99.2%
Boot Sectors	81.8%
Standard	100.0%
Polymorphic	100.0%

An extremely impressive set of results from *Avast!* puts the product well in the lead in terms of polymorphic virus detection. However, these excellent detection results mask a compatibility problem with the test machine: during scanning of the infected floppy diskettes, the machine would hang when the product was used in 'multiple floppy' mode.

AVScan v1.83

In the Wild	100.0%
Boot Sectors	90.9%
Standard	100.0%
Polymorphic	98.2%

H+BEDV's AVScan has always obtained good detection results, and this set is no exception, earning it a reputation as a reliable and easy-to-use scanner which gets the

job done. The impending release of an English language version of the company's commercial scanner may well provide another good choice for corporate use.

Central Point Anti-Virus v2.0

In the Wild	86.5%
Boot Sectors	72.7%
Standard	96.5%
Polymorphic	Failed to complete

It is almost becoming a feature of *Virus Bulletin* comparative reviews that *CPAV* is unable to complete the full set of tests. Although the problem of the product crashing after detecting 256 viruses has now been fixed, there are certain files which cause the machine either to pause for an unacceptably long time or to crash during scanning. The problem was repeatable, and needs to be solved.

Dr Solomon's AVTK v7.03

In the Wild	100.0%
Boot Sector	100.0%
Standard	100.0%
Polymorphic	99.5%

An excellent set of results from this well-respected product, missing only some samples of Cruncher and Smeg_v0.3. Scanning speeds on an infected machine are very slow, but clean disks are scanned at a respectable pace.

F-Prot Professional v2.14a

In the Wild	96.8%
Boot Sectors	100.0%
Standard	99.6%
Polymorphic	94.6%

Like so many other products, the more esoteric polymorphic viruses caused some problems for *F-Prot*. Despite this, the product still performed well overall, remaining close to the top of the field.

IBM Anti-Virus with PC-DOS v6.3

In the Wild	84.1%
Boot Sectors	63.6%
Standard	97.8%
Polymorphic	10.4%

Just like *MSAV*, the poor performance of this product is chiefly due to its age: the files on the disk were dated 25/01/94. Like several other products, *IBMAV* was unable to scan the Quox-infected diskette. Results for the commercial release of *IBM Anti-Virus* are given below.

IBM Anti-Virus v1.07

In the Wild	94.4%
Boot Sectors	100.0%
Standard	99.6%
Polymorphic	55.1%

Considerably improved scores from the version shipped with *PC-DOS v6.3*, although polymorphic virus detection is still lacking.

InocuLAN v3.0

In the Wild	89.7%
Boot Sectors	63.6%
Standard	97.4%
Polymorphic	12.6%

The workstation component of *InocuLAN* uses the same scanning engine as *Iris AntiVirus+*, and obtained exactly the same detection results. Although the product could detect the presence of the Mutation Engine, very few of the more recent and complex polymorphics were picked up by the scanner, an important consideration in a *NetWare* product.

McAfee Scan v2.1.1

In The Wild	90.5%
Boot Sectors	100.0%
Standard	96.1%
Polymorphic	57.4%

The new version of *McAfee Scan* looks more like a pre-release copy of the software rather than a finished product. After installation, the product refused to run, producing 'Error code 2' in 'Source ph_futil.c, Location: 1713, Status -1, Information $Revision: 1.15$ Data file not found.'

This was eventually tracked down to a problem with missing data files which the installation routine had inadvertently deleted from the disk. Surely an error message saying that simply and clearly would have been more appropriate?

The detection results are rather confusing, as the product would under certain circumstances detect a virus during the scan, but not include the virus in its summary of the scan results. This is an extremely serious bug, and one can only hope *McAfee Associates* has since fixed it. Version 2.1 of *Scan* may well be faster, but has a long way to go if it is ever to reach the top.

	In The Wild (126)	Boot Sector (11)	Standard (230)	Polymorphic (4796)	Overall (100)
AntiVirus+	114	7	224	602	66.0
Anyware AntiVirus	104	8	226	500	66.0
Avast!	125	9	230	4796	95.3
AVScan	126	10	230	4712	97.3
CPAV	109	8	222	Failed to complete	63.9
Dr Solomon's AVTK	126	11	230	4773	99.9
F-Prot Professional	122	11	229	4535	97.8
IBMAV with PC-DOS	106	7	225	500	64.0
IBMAV v1.07	119	11	229	2643	87.3
InocuLAN	113	7	224	602	65.8
McAfee Scan	114	11	221	2751	86.0
MSAV	77	2	210	466	45.1
Norman Virus Control	121	10	227	1586	79.7
Norton AntiVirus	109	6	225	1624	68.2
Novell DOS7	106	7	222	526	63.8
PCVP	117	7	220	2908	78.2
Scan Vakzin	95	8	217	561	63.5
Sweep	125	9	230	3743	89.8
ThunderBYTE	125	11	230	4704	99.3
VET	120	11	226	4728	98.0
Virus Alert	126	9	230	3550	89.0
Virus Buster	106	10	213	700	70.6
ViruSafe	102	9	222	1525	72.8
Vi-SPY	124	9	230	3668	89.2

Detection Results: This year's review has left room for every developer to improve: no-one escaped unscathed! Note that many of the viruses in the Polymorphic test-set are quite new. This therefore exposes those vendors who either update their products erratically, or take a long time to add new virus signatures. Another area where more products failed this year was the Boot Sector collection: only seven vendors scored 100%.

Microsoft Anti-Virus

In the Wild	61.1%
Boot Sector	18.2%
Standard	91.3%
Polymorphic	9.7%

Although some of the viruses missed by *MSAV* were due to the age of the product (the most recent file was dated 31/05/94), the poor detection results are still inexcusable. Despite the fact that the scanner is not the oldest in the test, *MSAV* has the dubious honour of being placed last.

Norman NVC v3.43

In the Wild	96.0%
Boot Sector	90.9%
Standard	98.7%
Polymorphic	33.1%

A disappointing score on the Polymorphic test-set mars what is otherwise a reasonable set of results for *Norman Data Defence System's NVC*. The only missed boot sector virus was Quox, where the product refused to scan the diskette in the drive, reporting 'No Floppy found in A:'.

	File Dates	Bernoulli Read (KB/sec)	Bernoulli Speed (min:sec)	Clean Diskette (min:sec)	Infected Diskette (min:sec)
AntiVirus+	19/10/94	20.7	61:09	0:52	3:33
Anyware AntiVirus	15/07/94	28.2	44:55	1:37	2:55
Avast!	14/10/94	15.1	83:42	1:17	3:00
AVScan	20/10/94	29.5	43:00	1:31	2:07
CPAV	23/09/94	Failed to complete	Failed to complete	2:15	3:50
Dr Solomon's AVTK	20/09/94	64.7	19:35	0:57	16:35
F-Prot Professional	30/09/94	66.0	19:13	1:13	2:45
IBMAV with PC-DOS	25/01/94	Failed to complete	Failed to complete	1:55	3:52
IBMAV v1.07	05/08/94	15.4	82:15	1:27	3:17
InocuLAN	11/08/94	58.7	21:35	0:55	4:57
McAfee Scan	17/10/94	36.6	34:39	0:48	4:35
MSAV	31/05/94	21.8	58:05	1:14	6:22
Norman Virus Control	10/09/94	59.0	21:28	1:22	2:15
Norton AntiVirus	03/02/94	76.1	16:40	0:40	2:27
Novell DOS7	26/01/94	36.6	34:40	1:27	7:54
PCVP	20/10/94	353.7	3:35	0:37	0:54
Scan Vakzin	11/10/94	74.4	17:02	1:11	3:06
Sweep	03/10/94	36.3	34:53	1:15	1:43
ThunderBYTE	24/10/94	603.6	2:06	0:23	2:18
VET	20/09/94	97.5	13:00	0:55	1:36
Virus Alert	27/09/94	585.0	2:10	0:24	2:21
Virus Buster	07/10/94	38.2	33:12	0:27	9:00
ViruSafe	18/10/94	65.3	19:25	0:38	1:35
Vi-SPY	20/10/94	32.7	38:47	1:05	6:25

Speed Results: Although speed is not everything, a scanner which is as unobtrusive as possible is certainly an advantage. Once again, *ThunderBYTE* from *ESaSS* streaked ahead of the rest of the field, clocking up a monumental scan speed of 603.6 KBytes per second, without compromising its detection rates.

Norton AntiVirus v3.0

In the Wild	86.5%
Boot Sector	54.5%
Standard	97.8%
Polymorphic	33.9%

The version of *NAV* sent in for review was quite old (dated 03/02/94), which goes some way toward explaining the product's poor detection results. Note, however, that *Virus Bulletin* has a strict policy of reviewing what it is sent: one hopes that buyers receive a more up-to-date copy.

Novell DOS 7

In the Wild	84.1%
Boot Sector	63.6%
Standard	96.5%
Polymorphic	11.0%

Like the other anti-virus products included with the operating system, *NDOS7's* scanner is fearfully out of date, and this is reflected in its detection results. If one is determined to use the protection provided with DOS, this result shows that it is vital to ensure it has been recently updated.

PCVP v2.06

In the Wild	92.9%
Boot Sector	63.6%
Standard	95.7%
Polymorphic	60.6%

Like many other scanners, *PCVP* was unable to scan the boot sector of the Quox-infected diskette. Improvements are needed in both the Polymorphic and Boot Sector test-sets.

Scan Vakzin v4.167

In The Wild	75.4%
Boot Sectors	72.7%
Standard	94.3%
Polymorphic	11.7%

Scan Vakzin is the first ever Japanese scanner to be entered into a *Virus Bulletin* comparative review. Boot sector virus detection was acceptable, although, like several other products, the scanner refused to recognise the Quox-infected diskette. More seriously, no errors were reported by the scanner, meaning that a user scanning several floppy disks might not be aware of the problem. Detection of polymorphic viruses was also lacking.

Sophos' Sweep v2.66

In the Wild	99.2%
Boot Sectors	81.8%
Standard	100.0%
Polymorphic	78.0%

A solid set of detection results by *Sophos' Sweep*, although the product did seem to have problems with some of the newer viruses in the polymorphic test-set, and was still unable to detect the sample of the Quox virus, which is known to be in the wild. Still a good product, but with room for improvement.

ThunderBYTE v6.26

In the Wild	99.2%
Boot Sectors	100.0%
Standard	100.0%
Polymorphic	98.1%

Another set of cracking detection scores from this speedy product - a combination of good detection and incredible scanning speed puts *ThunderBYTE* very close to the top of the pack.

VET v7.825

In The Wild	95.2%
Boot Sectors	100% - but see text
Standard	98.3%
Polymorphic	98.6%

A very good set of test results from this antipodean product. The only point to note is that two of the boot sector viruses (Natas and Peanut) were not detected as such, but caused *VET* to display the message 'VET does not recognise the boot sector on the disk. It is probably harmless, but it COULD contain a virus'. This provides a useful early-warning system for new viruses, but relies on the developers of *VET* maintaining a large collection of valid boot sectors.

Virus Alert v3.24

In the Wild	100.0%
Boot Sectors	81.8%
Standard	100.0%
Polymorphic	74.0%

Virus Alert, produced by *Look Software in Canada*, sports a bright cheerful user interface (right down to wishing the user 'Joy Peace and Happiness' on the closedown screen). In terms of both speed and detection, it performed extremely well, although polymorphic virus detection could still be improved. Well worth a second look.

Virus Buster v4.04.01

In the Wild	84.1%
Boot Sector	90.9%
Standard	92.6%
Polymorphic	14.6%

Leprechaun's Virus Buster does not have a clear default mode of operation. However, timings and detection results are given for the fast scan mode. If the secure scan is selected, only the detection for the standard test-set changes, rising to 93.5%. Polymorphic virus detection was poor.

ViruSafe v6.3

In the Wild	81.0%
Boot Sectors	81.8%
Standard	96.5%
Polymorphic	31.8%

A middle of the road set of detection figures from *EliaShim*, which leaves plenty of room for improvement.

Vi-SPY v12 rel 11.94

In the Wild	98.4%
Boot Sectors	81.8%
Standard	100.0%
Polymorphic	76.5%

Vi-SPY, by *RG Software*, is another product which has a history of scoring well in *VB* reviews. This year's results are no exception, although the product missed two viruses in the wild (Peanut and Phantom1) and two boot sector viruses (Peanut and Quox). The product's inability to detect the Quox virus stems from its refusal to scan the Quox-infected disk, baldly stating that it had encountered a 'general failure reading drive a:'.

Closing Thoughts

There is no question about this being the toughest *Virus Bulletin* Comparative Review ever carried out. The viruses used in the polymorphic test-set have been chosen carefully to include several new samples, in order to highlight those vendors which are keeping their products completely up to date. Similarly, the boot sector virus collection has been revamped in order to reflect the changing threat. These enhancements have stretched the field, leaving one clear winner in terms of detection.

Every product should detect 100% of the In the Wild test-set: no excuse should be accepted from any vendor who did not score highly in this test. In practice, however, only three products achieved perfect scores in this test.

Speed results are given for a number of different system setups. Note that all products were run wherever possible in their default modes, although for testing purposes certain options sometimes had to be selected.

Perhaps the most critical timing test was that on an uninfected diskette: this represents one of the most common uses of the scanner. Scan time under heavily-infected conditions is less critical, though the reader should bear in mind that, during a large virus outbreak, there are likely to be a number of infected executable machines which will need to be cleaned. Too long a scan time on such a machine could make the product unusable when it is most needed.

Several products have performed so well that they deserve an individual mention. The top product in terms of virus detection was *Dr Solomon's AntiVirus ToolKit*, followed closely by *ThunderBYTE*, the second most accurate scanner, and by far the fastest. Other products which are worthy of praise are *Cybec's VET*, and *Alwil Software's Avast!*, which gained an extremely impressive 100% against the tricky polymorphic test-set.

If your product is not one of these, and is not at least scoring highly in the Boot Sector and In the Wild test-sets, you should ask your vendor to explain why.

TEST-SETS:

Boot Sector Test-set: One genuine infection on HD 1.44M 3.5-inch diskette of:

Natas, Junkie, NoInt, Peanut, BFD-451, AntiEXE.A, Parity_Boot, Empire.Monkey, Form, Quox, and LZR.

Polymorphic Test-set. 4796 infections of:

Cruncher (25), Girafe (1024), Groove and Coffee_Shop (500), One_Half (1024), Pathogen (1024), Satan_Bug (100), SMEG_v0.3 (1024), Uruguay.4 (75)

In the Wild Test-Set. 126 genuine infections of:

4K (Frodo.Frodo.A) (2), Argyle, Athens (2), Barrotes.1310.A (2), BFD-451, Black_Monday (2), Butterfly, Captain_Trips (2), Cascade.1701, Cascade.1704, Chill, CMOS1-T1, CMOS1-T2, Coffeeshop (2), Dark_Avenger.1800.A (2), Dark_Avenger.2100.DI.A (2), Dark_Avenger.Father (2), Datalock.920.A (2), Dir-II.A, DOSHunter, Eddie-2.A (2), Fax_Free.Topo, Fichv.2.1, Flip.2153.E (2), Green_Caterpillar.1575.A (2), Halloechen.A (2), Helloween.1376 (2), Hidenowt, HLLC.Even_Beeper.A, Jerusalem.1808.Standard (2), Jerusalem.Anticad (2), Jerusalem.PcVrsDs (2), Jerusalem.Zerotime.Australian.A (2), Junkie, KAOS4 (2), Keypress.1232.A (2), Lamer's_Suprise, Liberty.2857.D (2), Loren (2), Macgyver.2803.B, Maltese_Amoeba (2), Natas, Necros (2), No_Frills.843 (2), No_Frills.Dudley (2), Nomenklatura (2), Nothing, Nov_17th.855.A (2), Npox.963.A (2), Old_Yankee.1, Old_Yankee.2, Peanut, Phantom1 (2), Pitch, Piter.A, Power_Pump.1, Revenge, Screaming_Fist.II.696 (2), Satan_Bug (2), SBC, Sibel_Sheep (2), Spanish_Telecom (2), Spanz, Starship (2), SVC.3103.A (2), Syslock.Macho (2), Syslock.Syslock.A, Tequila, Todor (2), Tremor (5), Vacsina.Penza.700 (2), Vacsina.TP.5.A, Vienna.627.A, Vienna.648.A, Vienna.W-13.534.A, Vienna.W-13.507.B, Virdem.1336.English, Warrier, Warrior, Whale, XPEH.4928 (2).

Standard Test-set. 230 genuine infections of:

1049, 1260, 12_Tricks, 1575, 1600, 2100 (2), 2144 (2), 405, 417, 492, 4K (2), 5120, 516, 600, 696, 707, 777, 800, 8888, 8_Tunes, 905, 948, AIDS, AIDS II, Alabama, Ambulance, Amoeba (2), Amstrad (2), Anthrax (2), AntiCAD (2), Anti-Pascal (5), Armagedon, Attention, Bebe, Blood, Burger (3), Butterfly, Captain_Trips (2), Cascade (2), Casper, Coffee_Shop, Dark_Avenger, Darth_Vader (3), Datalock (2), Datacrime, Datacrime_II (2), December 24th, Destructor, Diamond (2), Dir, Diskjeb, DOSHunter, Dot_Killer, Durban, Eddie, Eddie 2, Fellowship, Fish_1100, Fish_6 (2), Flash, Flip (2), Fu_Manchu (2), Halley, Hallochen, Helloween (2), Hide_Nowt, Hymn (2), Icelandic (3), Internal, Invisible_Man (2), Itavir, Jerusalem (2), Jocker, Jo-Jo, July_13th, Kamikaze, Kemerovo, Kennedy, Keypress (2), Lehigh, Liberty (5), LoveChild, Lozinsky, Macho (2), Maltese_Amoeba, MIX1 (2), MLTI, Monxla, Murphy (2), Necropolis, Nina, Nomenklatura (2), NukeHard, Number_of_the_Beast (5), Oropax, Parity, PcVrsDs (2), Perfume, Pitch, Piter, Polish 217, Power_Pump, Pretoria, Prudents, Rat, Satan_Bug (2), Shake, Sibel_Sheep (2), Slow, Spanish_Telecom (2), Spanz, Starship (2), Subliminal, Sunday (2), Suomi, Suriv_1.01, Suriv_2.01, SVC (2), Sverdlov (2), Svir, Sylvia, Syslock, Taiwan (2), Tequila, Terror, Tiny (12), Todor, Traceback (2), Tremor, TUQ, Turbo_488, Typo, V2P6, Vacsina (8), Vcomm (2), VFSI, Victor, Vienna (8), Violator, Virdem, Virus-101 (2), Virus-90, Voronezh (2), VP, V-1, W13 (2), Willow, WinVirus_14, Whale, Yankee (7), Zero_Bug.

PRODUCT REVIEW

Vi-SPY: Universal NIM?

Jonathan Burchell

So far, all the network anti-virus products tested in *Virus Bulletin* have been designed to execute on the network file server under the server operating system. Such a configuration has many advantages when compared with trying to produce network-wide protection using only workstation-based software. There is, of course, one major disadvantage: in most cases, the vendor supports only *Novell NetWare v3.11* or above.

RG Software System's VI-SPY Universal Network Installable Module (*NIM*) takes a different approach to network security. Rather than have a network operating system-dependent executable, the *NIM* is independent of underlying network technology, yet has some understanding of a network environment. There are many networks, such as *Banyan Vines*, for which no specific virus protection is readily available. Thus, a network-aware workstation-based solution may enable administrators of such networks to install an enterprise-wide virus protection scheme.

How it Works...

Network protection is based around three central issues: providing real-time protection from infection, providing scheduled scans of servers, and issues related to ease of administration, user management, and messaging/reporting. At some level, the *Vi-SPY NIM* tackles all these issues.

The core of the *NIM* is the *Vi-SPY* professional workstation product. This has been combined with a number of utilities and batch files designed to make it appropriate to a network environment. The *Vi-SPY* workstation product has already been reviewed in *VB* (June 1994, p.20): I will state simply that is a typical DOS Scanner and TSR, with above average detection rates.

Vi-SPY provides integrity through checking of files, partition tables, CMOS RAM and memory maps, and through detection of changes in files. The *NIM* does not have a user interface as such, making it rather difficult to review, so I have tried to describe a typical installation process instead.

Getting Started

The product consists of a single 3.5-inch high-density diskette (5.25-inch diskettes are available on request), a network administrator's guide, and a copy of the *Vi-SPY Professional* manual.

The software licence allows protection of a single server and of all workstations connected to that server, but excludes stand-alone operation. The issue of nomadic users is therefore rather unclear: many products allow protection of laptop and in some case 'home' PCs, recognising that the user who occasionally connects to the network, or the employee transporting disks between work and home, is a potential source of infection. [*RG Software has since stated that home machines are covered at no extra charge. Ed.*]

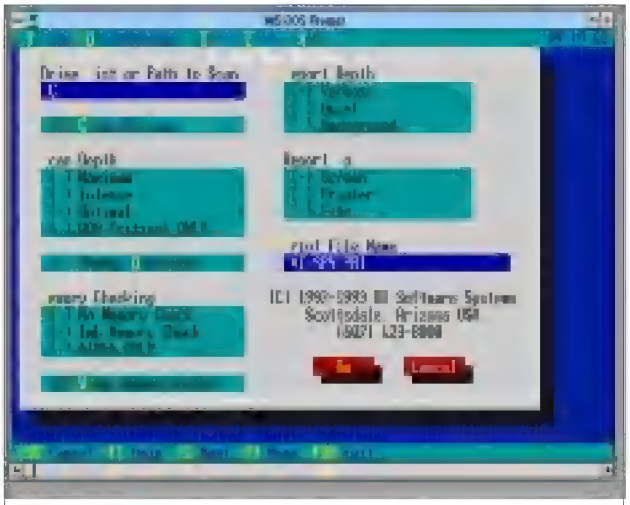
Vi-SPY is designed to be network-independent; thus, most of the installation and setting-up process involves customising batch files and program options to conform to one's networking environment. Ideally, in order to minimise the work in maintaining individual workstations, all *Vi-SPY* software is loaded from a network drive, and executed remotely on users' machines.

The best time to load protection software is before the user has had a chance to log on to the network and map any drives. *RG Software* suggests placing the *Vi-SPY NIM* files on a drive available prior to login (such as *Novell's* F:\LOGIN directory).

This will indeed work, but extra care must be taken in multiple-server environments. *Banyan Vines*, for instance, will allocate a user's login drive (known as drive Z) across available servers according to demand. With *Vines*, a user's login and logout drives may be allocated from different servers. This can cause particular problems with the execution of batch files, so care must be taken that all files on all Z: drives are identical.

Installation

The *NIM* is installed in a two-part operation: firstly, the software is loaded to a local hard disk; secondly, after the local workstation has been checked, its files are copied to



The *Vi-SPY Universal NIM* takes a different approach to other network-based anti-virus countermeasures by providing a highly-configurable network-independent package.

the network drive. On my machine, the procedure functioned smoothly, with the DOS software being loaded first, followed by the *MS-Windows* elements.

The installation process starts *Windows* to load elements pertaining to that application, but in my tests, when this happened, I was greeted by an error message informing me that the DOS TSR was not loaded. This was inevitable: *Vi-SPY* was still being installed, and the machine would need to be rebooted to install the TSR. *RG Software* should address this issue, as error messages during installation do not inspire user confidence [*RG Software claims that it is aware of this problem, and has fixed it in the shipping release. Ed.*]. The program also modified AUTOEXEC.BAT to check whether to run *Vi-SPY* and load the TSR.

When the test machine was rebooted, *Vi-SPY* ran, followed by the TSR. Unfortunately, by this stage my system had become completely unstable, and executing almost any command caused the system to hang. I suspected the real-time checking TSR (RVS): indeed, removing it made the system stable again. RVS has the ability to hold the virus signature and rules database in EMS or in XMS memory, using EMS memory by default. Changing that to XMS memory also stabilised the system again.

My test system utilises the *QEMM* memory manager fully, loading SCSI hard disk, Optical and CD-ROM drivers into high memory, as well as creating extra high memory via *QEMM's* stealth feature, which 'hides' ROMs in extended memory, bringing them back during a ROM interrupt.

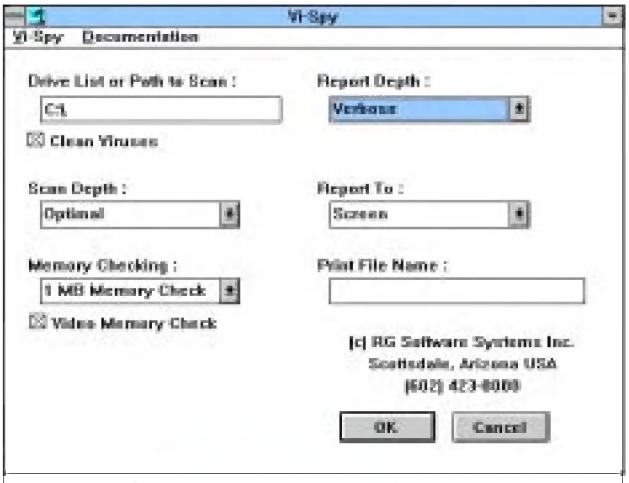
This system is exceptionally stable, but the stealth feature allows the software to carry out only 'legitimate' actions. I suspect that the system instability problems I experienced were due to RVCS attempts to ensure it was still in control, and had not been subverted by virus code and stealthing.

The manual suggests that, having installed *Vi-SPY* onto a single workstation and virus-checked the local and remote drives, the *Vi-SPY* files are copied to the chosen network directory. The COPYVS batch file helps this process, but may require editing before use: for instance, the destination is hard-coded into the batch file as F:\LOGIN\VISPY.

Loading the NIM

The *Vi-SPY NIM* may be loaded either at AUTOEXEC.BAT time (which is why it has to be placed on a publicly-accessible drive) or at login time. Better protection is probably provided by loading it at AUTOEXEC time; however, this will mean modifying every AUTOEXEC.BAT file on the network. That would require a physical visit to every workstation (unless you have remote control software such as *PC-Anywhere*, or another means of automatically updating workstation drives).

The INST-NET program helps automate this task by displaying the workstation's AUTOEXEC.BAT file, allowing specification as to where a 'standard' set of



Vi-SPY proved to be no slouch in terms of detection or speed, beating the detection results of several better-known NLMs

command lines for invoking *Vi-SPY* should be inserted. The actual command lines are held in a separate text file, which can be altered with a text editor. An example of a standard set of commands might be:

- Run the VSUPDATE program, which automatically updates workstation *Vi-SPY* files from the server.
- Run a complete check of the workstation as specified by AutoVS. This program allows specification of checking frequency in terms of days, day of the week, or time since the last full check. This means a user is not 'inconvenienced' by a full check every time he reboots or logs in.
- Install the TSR.

The final task adds a call to the VSEXIT batch file into the login exit processing script, which ensures that, when a user logs on to the server, the VSEXIT batch file is run.

The sample VSEXIT batch file checks that *Vi-SPY* has been successfully run, as specified by AutoVS. If it is time for a full check, *Vi-SPY* will be executed to check users' local drives. Assuming no virus is detected, the batch file checks that the real-time monitoring TSR is loaded, locking the user workstation if it is not. The procedure for invoking *Vi-SPY* and RVCS from a login script is similar, although a slightly different VSEXIT batch file is used.

Vi-SPY and its companion TSR take many options which modify their operation. These may be specified on the command line, or in an INI file. Virus incidents can be reported to a shared centralised log file.

Once these loading tasks have been completed, a set-up will have been established which provides real-time checking of all files accessed from a workstation, together with complete periodic scans of the workstation via *Vi-SPY*.

If this all sounds rather handcrafted, that's because it is. *RG Software* provides a collection of tools which enables automation of the protection process in a networked environment. The approach taken to 'network independ-

ence' is to limit actions to operations which can be carried out via programs and batch files executed on a workstation. This requires both installer and administrator to have a good understanding of the environment, including drive mapping issues, batch file processing, and the login/logout process.

Scheduled scanning is effected through having a workstation running a slightly different form of the AutoVS scheduler, which is capable of invoking a command at a specified time. The workstation must be left at the DOS prompt for this to work, and must mount the server drives to be scanned. In view of the requirement for the workstation to be idle between tasks, and of the security implications, it would probably be best to dedicate a PC to this task, and place it in a physically secure location.

Results

Vi-SPY and the real-time scanner are very good at detecting viruses: the only virus samples missed from the test-set were the DIET-compressed Cruncher infections. One of the most remarkable things about the real-time TSR is the overhead figure. The TSR seems to add less than 20% overhead to file I/O: this is highly commendable.

Conclusions

The *NIM* extends its capabilities to a networked environment, but even given its excellent detection results, I would not recommend its use in preference to a server-based solution if one of an equivalent detection standard exists (for a comparison, see *VB*, October 1994, pp.13-20). The facilities for administration, reporting and messaging simply do not compare with the server-based solutions; installation is relatively complex, and there are many issues related to having the software workstation-based.

There are, however, environments where server-based solutions are not available: where that is the case, this product would provide an extremely useful collection of utilities, programs and batch files, and would allow a form of automated network-wide protection to be established.

RG Software needs to extend the product in two areas: firstly, to provide more set-up and configuration assistance, a menuing interface needs to be added to help automate the generation of the controlling environment. Secondly, reporting and messaging facilities need to be extended. Reporting is really non-existent in the current product, and there is no provision for messaging. Facilities should be added which allow network messages to be sent in the case of infection or other problems. Most network systems already have some sort of 'SendMessage' utility - the company should perhaps explain how to utilise this feature.

Once the product becomes more 'network-friendly', and the capabilities discussed above are added, *Vi-SPY's* NIM will be an invaluable addition to any network environment, and able to compete strongly. Even without these enhancements, it is still well worth considering for unusual networks.

Vi-SPY

Detection Results:

Main Scanner:

Standard Test-Set[1]	229/229	100%
In the Wild Test-Set[2]	109/109	100%
Polymorphic Test-Set[3]	576/600	96.0%

Detection results for real time checking via the TSR are identical.

Overhead of real-time TSR:

Time to copy 1071 files	
Without scanner	66 seconds
With scanner	79 seconds

Technical Details

Product: *Vi-SPY Netware Installable Module.*

Developer: *RG Software Systems*, 6900 East Camelback Road, Suite 630, Scottsdale, AZ 85251, USA. Tel. +1 602 423 8000, Fax +1 602 423 8389.

Price: US$995 for a single copy; includes unlimited workstations attached to that server. Also includes quarterly updates; monthly updates available free of charge through downloading from the company BBS.

Hardware used: Client machine - 33 MHz 486, 200 Mbyte IDE drive, 16 Mbytes RAM. File server - 33 MHz 486, EISA bus, 32-bit caching disk controller, *Novell NetWare version 3.11*, 16 Mbytes RAM.

Each test-set contains genuine infections (in both COM and EXE format where appropriate) of the following viruses:

[1] **Standard Test-Set:** As printed in *VB*, February 1994, p.23 (file infectors only).

[2] **In the Wild Test-Set:** 4K (Frodo.Frodo.A), Barrotes.1310.A, BFD-451, Butterfly, Captain_Trips, Cascade.1701, Cascade.1704, CMOS1-T1, CMOS1-T2, Coffeeshop, Dark_Avenger.1800.A, Dark_Avenger.2100.DI.A, Dark_Avenger.Father, Datalock.920.A, Dir-II.A, DOSHunter, Eddie-2.A, Fax_Free.Topo, Fichv.2.1, Flip.2153.E, Green_Caterpillar.1575.A, Halloechen.A, Helloween.1376, Hidenowt, HLLC.Even_Beeper.A, Jerusalem.1808.Standard, Jerusalem.Anticad, Jerusalem.PcVrsDs, Jerusalem.Zerotime.Australian.A, Keypress.1232.A, Liberty.2857.D, Maltese_Amoeba, Necros, No_Frills.843, No_Frills.Dudley, Nomenklatura, Nothing, Nov_17th.855.A, Npox.963.A, Old_Yankee.1, Old_Yankee.2, Pitch, Piter.A, Power_Pump.1, Revenge, Screaming_Fist.II.696, Satanbug, SBC, Sibel_Sheep, Spanish_Telecom, Spanz, Starship, SVC.3103.A, Syslock.Macho, Tequila, Todor, Tremor (5), Vacsina.Penza.700, Vacsina.TP.5.A, Vienna.627.A, Vienna.648.A, Vienna.W-13.534.A, Vienna.W-13.507.B, Virdem.1336.English, Warrior, Whale, XPEH.4928

[3] **Polymorphic Test-Set:** 600 genuine samples of: Coffeeshop (250), Groove (250), Cruncher (25), Uruguay.4 (75).

NB: Note that the test-sets used in this review are different from those used in the DOS Scanner Comparative Review (pp.14-19). The virus test-sets used for NLM evaluations will be updated next month.

CONFERENCE REPORT

EICAR '94: Success in St. Albans?

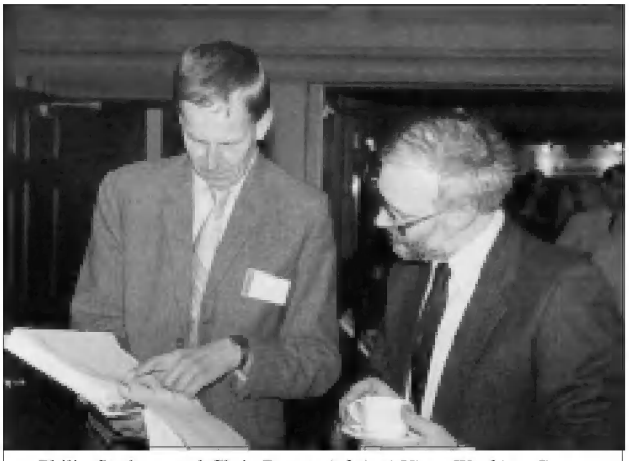
Perhaps the most enjoyable thing about being Editor of *Virus Bulletin* is the opportunity to travel to far-off and mysterious places. It was therefore something of a disappointment to discover that this year's *EICAR* (*European Institute of Computer Anti-virus Research*) conference was to be held in sunny St. Albans, only an hour's drive from the *VB* offices. How misconceived one can be!

Opening Night

Speakers at the conference and *EICAR* members were treated to a reception dinner, the evening before the conference proper. Held at the *Sopwell House Hotel* (also the conference venue), the evening passed most enjoyably, giving everyone a chance to renew old acquaintances and make new ones under informal conditions.

The following morning, delegates (some still fuzzy from the night before) gathered for an opening presentation by Michael Freiberg (*BP Oil*, Germany) on the role of incident reporting within a corporate IT environment. After this keytone talk, the day split into a coporate and a technical stream, featuring such well-known speakers as Alan Solomon, Vesselin Bontchev and Sara Gordon.

One of the most enjoyable papers of the day was presented by Kari Laine (*LAN Vision Oy*, Finland) on the 'cult' of anti-virus testing [*a subject close to my heart. Ed.*]. Laine pointed out the traps and pitfalls of product reviews, and even included a step-by-step guide on how to fix your own anti-virus tests! He believes that there is a lack of quality reviews and quality reviewers, and hopes that the ITSEC will change this. Only time will tell, but experience says that there will always be someone who is unhappy with the outcome of an anti-virus software evaluation. Combining this with the issue of testing a check-summer or a behaviour blocker, the true scale of the problem becomes apparent.



Philip Statham and Chris Baxter (of *Anti-Virus Working Group* fame), discussing the finer points of product evaluation.

Onward

The second day opened with a lecture from Fridrik Skulason on the impact of the virus flood. The conference then divided, the technical stream beginning with Chris Fischer's talk on computer virus analysis. He was followed by Franz Veldman, of *ESaSS BV*, who discussed the symbiosis between virus writers and anti-virus product developers.

The final technical paper came from Dmitri Gryaznov, of *S&S*, who spoke on *SimBoot*, a device which simulates both a diskette and user dialogue with a scanner. The process, uses no real diskettes, and is automatic, taking a fraction of the time needed for a manual scan. Gryaznov's talk included a practical demonstration: physically testing four scanners against five diskettes took approximately four minutes, and using *SimBoot* to test more than fifty simulated floppies against the same scanners took 3 minutes 10 seconds.

On the corporate side, a presentation by Sara Gordon (better known for research into motivations of virus writers), gave an interesting talk on computer viruses as cause for concern.

EICAR 94 closed with a panel session which put forward the question of what a sensible anti-virus strategy for the year 2000 might be. A general consensus, of the audience and the panel, was that there will be fewer anti-virus products, more integration of different types of software, and that products which do survive will be monolithic, unable to function efficiently in the morass of viruses expected by then.

In With the New

Also at *EICAR 94* was the controversial figure of virus author Tim Gaskin, *aka* Ice 9. He talked about his contact with *ARCV*, and discussed his decision to stop writing viruses, which came with the realisation of the damage and trauma he could cause other users. The audience, despite their preconceptions, listened carefully to his views.

The conference also saw the founding chairman of *EICAR*, Dr Paul Langemeyer, step down to make way for new blood. A new board has been elected, with Michael Freiberg in the chair, and it hopes to attract more new members.

Congratulations to Julie Bartle and her team at *S&S International* for the excellent organisation, which added polish to the proceedings. *EICAR 95* will be held in Zurich, under the auspices of the Swiss-based *CIMA AG*. Will it be worth the trip? One can only quote from the movies: 'I'll be back'.

SUBSCRIPTION RATES

Subscription price for 1 year (12 issues) including first-class/airmail delivery:

UK £195, Europe £225, International £245 (US$395)

Editorial enquiries, subscription enquiries, orders and payments:

Virus Bulletin Ltd, 21 The Quadrant, Abingdon, Oxfordshire, OX14 3YS, England

Tel. 01235 555139, International Tel. +44 1235 555139
Fax 01235 531889, International Fax +44 1235 531889
Email virusbtn@vax.ox.ac.uk
CompuServe 100070,1340@compuserve.com

US subscriptions only:

June Jordan, *Virus Bulletin*, 590 Danbury Road, Ridgefield, CT 06877, USA

Tel. +1 203 431 8720, Fax +1 203 431 8165

END NOTES AND NEWS

This year's *Virus Bulletin* conference, *VB 95*, will be held at the *Boston Park Plaza Hotel* from 20-22 September 1995, in Boston, Massachusetts, USA. Anyone wishing to submit an abstract for consideration for the conference should do so by 31 January 1995. Further details on this, and on any aspect of *VB 95*, can be obtained from conference manager Petra Duffield, Tel. +44 (0)1235 555139, fax +44 (0)1235 531889.

***VSUM* scanner listings for November 1994** (month in brackets indicates when that version of the product was certified by *VSUM*):
DOS-based products: 1. *McAfee Associates ViruScan v2.1.3*, 97.5% (9411), 2. *Dr Solomon's AVTK v 6.69*, 96.4% (9411), 3. *Sophos' Sweep v2.67*, 96.0% (9411), 4. *Command Software's F-Prot Professional 2.13*, 95.8% (9406), 5. *IBM Anti-Virus for DOS v1.07E*, 92.2% (9411).
NLMs: 1. *Sophos' Sweep v2.67*, 96.8% (9411), 2. *Dr Solomon's AVTK v6.66*, 92.9% (9411), 3. *McAfee NetShield v1.6v117*, 91.1% (9408), 4. *Command Software's Net-Prot v1.25*, 81.0% (9406).

The *EUROSEC 95 Forum* will take place from 23-25 March 1995 at the *Hotel Lutetia*, Paris, France. The conference covers many aspects of IT security, including identification of future developments and requirements in the area. Details from Isabelle Hachin at *XP Conseil*. Tel. +33 (1) 42 89 65 65, fax +33 (1) 42 89 65 66.

Another contribution to the information security market has been launched recently by Scottsdale AZ (USA)-based *M&T Technologies*. *MicroSAFE Laptop* was developed to counter the threat posed by the loss or comprimisation of information stored on laptops. The product offers **access control, virus protection, and information security for portable computers.** The system is available for circa US$50.00 per unit. Michael Pressendo, of *Gordon C James Public Relations*, is available to answer queries: Tel. +1 602 274 1988; fax +1 602 274 2088.

Precise Publishing has launched **a new disk authorisation package** called *The Enforcer*. Other facilities offered by the package include writing a clean boot sector to authorised floppies, optional boot-up passwords, and disk locking. Free evaluation copies are available on request. Tel. +44 (0)1384 560527.

The **Anti-Virus Workshop from *Sophos*** on 25/26 January 1995 at the training suite in Abingdon has three places free, but only on the 'Advanced' day. The next dates are 29/30 March. To reserve a place, or to obtain information, contact Karen Richardson at *Sophos plc* on Tel. +44 (0)1235 559933, fax +44 (0)1235 559935.

The US Government's *NIST* (*National Institute of Standards and Technology*) has released a CD-ROM containing such things as complete issues of the computer underground magazine *Phrack*, postings from *Virus-L*, a series of password dictionaries, and (possibly the biggest collection on the disk) **anti-virus tools and techniques**. Contact *NIST* directly on Tel. +1 301 965 3240 for information.

ESaSS BV, producer of *ThunderBYTE*, has moved. Effective immediately, the **new contact address** is: Saltshof 10-18, NL-6604 EA WIJCHEN, The Netherlands. The new numbers are:
Tel. +31 8894 2282, fax +31 8894 50899. NB: UK users can still contact *ESaSS'* UK distributor, *Reflex Magnetics*, on its usual telephone number: 0171 372 6666.

A federation of Japanese computer clubs is planning to present the US president, Bill Clinton, with **a harmless new computer virus** designed to teach about the danger of computer viruses. The program does not erase data from the computer's memory, and still allows use of the computer's word processing and calculating functions. The virus, Kyoto Ichigo (Kyoto Number One) can only currently be used on NEC PCS, but work is underway to make it function on *IBM* PCS and *IBM* compatibles. [*It's a mad world... Ed.*]